

Beautiful Structures: An Appreciation of the Contributions of Alan Selman*

Lane A. Hemaspaandra[†]
Department of Computer Science
University of Rochester
Rochester, NY 14627, USA

June 17, 2014

Abstract

Professor Alan Selman has been a giant in the field of computational complexity for the past forty years. This article is an appreciation, on the occasion of his retirement, of some of the most lovely concepts and results that Alan has contributed to the field.

1 Preface



Figure 1: Alan Selman

As I write these words in June 2014, it has been just over a month since the retirement celebration for Alan Selman at the University at Buffalo's Center for Tomorrow. I can't think of a more fitting location for the celebration, given that Alan's technical contributions

*This appreciation will appear, in slightly different form, in the Complexity Theory Column of the September 2014 issue of *SIGACT News*.

[†]Supported in part by grant NSF-CCF-1101479.

to the field are of such beauty and insight that they are as important to the field's future as they have been to its past and present.

Any retirement is bittersweet, but Alan has mentioned that he will be keeping his hand in the field in retirement. That happy fact helped all of us at the celebration focus on the sweet side of the bittersweet event. Warm talks and memories were shared by everyone from the university's president, the department chair, and Alan's faculty colleagues all the way up to the people who are dearest of all to Alan—his postdocs and students.

The warmth was no surprise. Alan is not just respected by but also is adored by those who have worked with him. Anyone who knows Alan knows why. Alan is truly kind, shockingly wise, and simply by his nature devoted to helping younger researchers better themselves and the field. But in fact, I think there is more to say—something far rarer than those all too rare characteristics. What one finds in Alan is a true belief in—an absolute, unshakable belief in—the importance of understanding of the foundations of the field.

Now, one might think that Alan holds that belief as an article of faith. But my sense is that he holds the belief as an article of understanding. Like all the very, very best theoreticians, Alan has a terrific intuition about what is in the tapestry of coherent beauty that binds together the structure of computation. He doesn't see it all or even most of it—no one ever has. But he knows it is there. And in these days when many nontheory people throw experiments and heuristics at hard problems, often without much of a framework for understanding behaviors or evaluating outcomes, not everyone can be said to even know that there is an organized, beautiful whole to be seen. Further, Alan has such a strong sense for what is part of the tapestry that—far more than most people—he has revealed the tapestry's parts and has guided his collaborators and students in learning the art of discovering pieces of the tapestry.

And that brings us to the present article and its theme of the beauty of the structures and the structure that Alan has revealed—the notions, the directions, and the theorems. For all of us whose understanding isn't as deep as Alan's, the beauty of Alan's work has helped us to gain understanding, and to know that that tapestry really is out there, waiting to be increasingly discovered by the field, square inch by square inch, in a process that if it stretches beyond individual lifetimes nonetheless enriches the lifetimes of those involved in the pursuit of something truly important. To summarize Alan's career in a sentence that is a very high although utterly deserved compliment: Alan is a true structural complexity theorist.

2 Introduction

It would be impossible to cover in a single reasonably sized article all or even most of Alan's contributions to the field. So this article will celebrate Alan's career in a somewhat unusual way. Given that the heart of Alan's contribution to the field is truly beautiful structures— notions, directions and foundational results regarding those—this article will simply present to the reader a few of those structures and point out (when it isn't already apparent) why they are beautiful and important.

We won't be trying to survey the results that are by now known about the structures, although we'll often mention the results that Alan and his collaborators obtained on the structures in their original work, and we'll sometimes mention some later results. But the core goal here is to present the beautiful structures themselves.

We'll do that in Section 3. But before turning to that, it would be a crime not to at least allude to the stunning breadth of Alan's contributions in service and developing human infrastructure, and to the recognition he has received for his scholarship and service.

Alan has trained many Ph.D. students (his already graduated Ph.D. advisees are Joachim Grollmann, John Geske, Roy Rubinstein, Ashish Naik, Aduri Pavan, Samik Sen-gupta, and Liyu Zhang, and his current Ph.D. advisees are Andrew Hughes, Dung Nguyen, and Nathan Russell) and postdocs (Mitsunori Ogihara, Edith Hemaspaandra, and Christian Glaßer), and their contributions—both under Alan and beyond—have been important to the field. Those of us who were not his students or postdocs but have had the privilege of working with Alan have been enriched, inspired, and uplifted by his insights and vision. The field and his school have recognized Alan's research and service contributions with a long list of the highest awards and most important positions. Alan is a Fellow of the ACM, a Fulbright scholar, and the recipient of an Alexander von Humboldt Foundation Senior Research Award and a Japan Society for the Promotion of Science Invitational Fellowship. He has been awarded the ACM SIGACT Distinguished Service Prize, and has a long record of extensive editorial service to the field, including being the editor-in-chief of *Theory of Computing Systems* since 2001. He is the recipient of the State University of New York Chancellor's Award for Excellence in Scholarship and Creative Activities, and of the University of Buffalo's Exceptional Scholar Award for Sustained Achievement. Alan was acting Dean of the College of Computer Science at Northeastern University, and chaired UB's Computer Science and Engineering department for six years. With Steve Homer, Alan wrote a computability and complexity textbook [HS11], and he edited the *Complexity Theory Retrospectives* [Sel90, HS97]. Alan was an important part of efforts to obtain stronger government grant support for the study of theoretical computer science. He was instrumental in the creation of, and served the first term as conference chair of, the IEEE Conference on Structure in Complexity Theory (now called the IEEE Conference on Computational Complexity), and for that won the IEEE Computer Science Society Meritorious Service Award.

That's some record!

3 Beautiful Structures

In this article, we'll present five of Alan's beautiful notions—concepts that Alan has introduced or very substantially advanced. That is such a small number relative to the dozens of topics that Alan has contributed to that we'll be skipping topics that for almost any other person would in and of themselves be the highlight of an entire career. In fact, we won't even be trying to pick off the “top five” notions, but rather will just be selecting a set of five very lovely notions.

For example, we'll skip right over the seismic contribution of Ladner, Lynch, and Selman [LLS75] to the definitions of and understanding of the relative powers of the rich range of polynomial-time reducibilities that are so widely used today.

And similarly, since Alan and his collaborators will soon be surveying issues related to these topics in a *SIGACT News* Complexity Theory Column [GHSW14], we also will leave untouched all issues related to disjoint pairs, promise problems, and propositional proof systems, even though Alan's importance there is great, stretching across more than a half dozen papers, from the seminal 1980s work of Even, Selman, and Yacobi [ESY84] to the remarkable 2010s work of Hughes, Pavan, Russell, and Selman [HPRS12].

Alan and his collaborators have in recent decades resolved a long-open issue, bringing unity to the understanding of mitoticity, completeness, and autoreducibility for many central complexity classes. Among their advances is that every (many-one polynomial-time) NP-complete set is (many-one polynomial-time) autoreducible, and (many-one polynomial-time) autoreducibility and (many-one polynomial-time) mitoticity coincide. So, for example, every NP-complete A set is so repetitively structured that there exists a P set B such that $A \cap B$ and $A \cap \overline{B}$ are infinite and (many-one polynomial-time) equivalent, and so certainly are themselves each (many-one polynomial-time) NP-complete. Also, Alan and his collaborators have brought great light to the extent to which this type of behavior persists or fails for other types of reductions, such as for reductions more flexible than many-one reductions, or reductions in the logspace world. And yes, you've guessed it, we also won't be covering any of that work here, since happily Alan and his collaborators half a decade ago wrote for the *SIGACT News* Complexity Theory Column a survey article on the work in this line up to that point [GOP⁺09], although if you are interested (as well you should be!), please don't miss their recent work on this line in ICALP-13 and STACS-14 [GNR⁺13, NS14].

$\text{NP} \cap \text{coNP}$ —not just itself, but as it functions when accessed through relativization (e.g., $\text{P}^{\text{NP} \cap \text{coNP}} = \text{NP} \cap \text{coNP}$ and $\text{NP}^{\text{NP} \cap \text{coNP}} = \text{NP}$), and in its cousin known as “strong” computing, and in the so-called strong nondeterministic reductions [Sel78, Lon82] based on that cousin—plays an important role in complexity theory. Alan was a central player in the key early work that built this collection of concepts [Sel74, Sel78, Sel79, Bra79, Lon82, Sch83]. And we won't cover that here.

We also won't cover what is now called the left-set technique, which was created by Alan [Sel88] and which for example was used to devastating effect by Ogihara and Watanabe [OW91] in their work showing that if any NP-complete set polynomial-time bounded-truth-table reduces to a sparse set, then $\text{P} = \text{NP}$.

There are many other beautiful themes, notions, and results in Alan's work, which we not only won't cover below but which we also haven't mentioned above. In fact, it should already be clear what we won't cover about Alan's work is enough to fill three or four extremely satisfying, productive, important careers. But were we to go on listing the important and lovely notions due to Alan that we *won't* cover, there would be no room left to actually cover any structures that Alan developed. So let us move right on to our select five.

As a reminder, for these notions we won’t at all be doing a survey of what is known, but rather we will be trying to convey what the notion is, why it is beautiful, and why its introduction by Alan and his collaborators was important. And a shorthand, this article will sometimes say “Alan and his collaborators” or even perhaps “Alan” when speaking of work by Alan joint with others; this is not in any way meant as a slight to those collaborators, but is simply since the focus of this article is on Alan. The citation labels (e.g., “[BLS84]”) will generally make it clear to the reader when we are employing this shorthand.

3.1 P-Selectivity: Choosing Which Path to Take When the Road Forks

Search issues are important not just in the area of Theory. AI researchers also are intensely focused on how to explore spaces.

So suppose you come to a fork in the road. There might be a treasure down one or both of the paths—or neither might hold a treasure. Which way should you go?

One of Alan’s beautiful structures addresses the issue of when polynomial-time functions can help you solve the above problem. After all, you don’t really need to know whether a given road has a treasure. That would be great to know, but maybe it is too much to hope for or too computationally expensive. Happily, all you really need to be able to do in the above situation is to choose one of the roads such that it is true that *if either of the roads holds a treasure, then the road you choose holds a treasure*.

In a quite amazing series of papers [Sel79, Sel81, Sel82a, Sel82b], Alan introduced and broadly explored the notion of P-selectivity, which captures precisely the above issue. We all know what polynomial time (P) is. A set A is in P exactly if there is a polynomial-time machine that accepts on each string in A and rejects on each string in \bar{A} . That is, one has a polynomial-time decision algorithm for membership in the set. For P-selectivity, one is required to have a polynomial-time semi-decision algorithm for the set. In particular—and here we use a particular one of the various equivalent definitions of this quite robust concept—a set A is said to be P-selective exactly if there is a polynomial-time function f that takes as its input two arguments, x and y , and has the property that, for each x and y ,

$$f(x, y) \in \{x, y\} \wedge (\{x, y\} \cap A \neq \emptyset \implies f(x, y) \in A).$$

What this says is that a set is P-selective exactly if there is a polynomial-time function that, given any two elements, always chooses one of them, and if at least one of them is in the set, the one it chooses is in the set. One sometimes hears this described by saying that the function chooses the element “more likely”—or, better, “no less likely”—to be in the set. This is a fine description, at least in the “no less likely” version, as long as one keeps in mind that the probabilities here are all zero and one.

P-selectivity is capturing the notion of wise search—being able to decide which way to go at forks in the road. It also is one of a great variety of notions (such as Almost Polynomial Time [MP79], P/poly [KL80], P-closeness [Sch86], Near Testability [GHJY91], Nearly Near Testability [HH91], etc.) that try to capture a wider range of sets than P does, yet to still have some natural polynomial-time action at their core.

Although this lovely notion, P-selectivity, was introduced by Alan, Alan’s seminal papers are quite open as to his inspiration. The P-selective sets (which are also sometimes called the semi-feasible sets, since they are the sets having polynomial-time semi-decision algorithms) were inspired by Jockusch’s notion from recursive function theory of the semi-recursive sets [Joc68]. In his career, Alan has often drawn on his broad understanding of recursive function theory to improve computer science’s pursuit of complexity-theoretic insights. This is of great value, given that so much of complexity theory—from the polynomial hierarchy [MS72, Sto76] to reductions to completeness to the Isomorphism Conjecture to immunity and bi-immunity to oracles and much more—is inspired by recursive function theory.

The present article focuses on presenting the beautiful structures themselves, rather than surveying everything known about them. But in this case it is important to note that although the P-selective sets have been extensively studied since Alan’s original series of papers (and indeed, there is even a monograph completely devoted to selectivity theory [HT03]), Alan’s original series of papers already went breathtakingly far in exploring this concept.

For example, Alan’s original papers already proved that there are P-selective sets that are not in P. Indeed, he (see also Ko [Ko83] regarding a more flexible type of left cut) showed that the left cut set (don’t worry if you don’t know what that term means) of any real number is P-selective. It follows from this that there are arbitrarily hard P-selective sets, e.g., there are P-selective sets that are so wildly undecidable that they aren’t even in the arithmetical hierarchy.

Alan also showed that if even one NP-hard set is P-selective, then $P = NP$. The proof is a lovely application of the self-reducibility of satisfiability. Going back to the parallel with wise search, this basically says that for NP-like search spaces that can be naturally cut in half repeatedly, having a P-time wise search algorithm is just too much to hope for. The proof is so crisp that it is worth sketching here. Suppose that some NP-hard set A is P-selective. Here is a polynomial-time algorithm for satisfiability. Given a boolean formula whose satisfiability we want to test, set its lexicographically first variable to true, get the resulting formula, and by A ’s NP-hardness transform that into a question about A . Do the same for our original formula with its first variable set to false. Take the outputs of these two processes, and use them as the inputs to the hypothesized polynomial-time P-selector function for A . That function will output one of them, and based on that choice, we’ll know either “If the formula is satisfiable then there is a satisfying assignment in which the first variable is set to true,” or “If the formula is satisfiable then there is a satisfying assignment in which the first variable is set to false.” So we take whichever assignment was just suggested to us by the selector, and we stick with it and similarly assign the second variable, again using the selector to fix it as being true or false, and so on until all the variables are assigned. At the end, we get to a single assignment such that if there is any assignment that satisfies the original formula, then *that* assignment satisfies the original formula. So we see if that one assignment satisfies the formula. If it does the formula is satisfiable, and otherwise the formula is not satisfiable. Lovely.

The results mentioned above are just a few examples of the broad investigation Alan carried out.

The facts that P-selective sets can be arbitrarily hard, and can't be NP-hard unless $P = NP$, might lead one to say that this class is terribly far beyond P. However, using a tournament-inspired divide-and-conquer framework, Ko [Ko83] proved that each P-selective set is information-wise very close to P. In particular, for each P-selective set there is a polynomial-time algorithm that given a polynomial amount of extra information based only on the *length* of the input string can correctly accept the set. In the jargon, each P-selective set is in the complexity class P/poly, or equivalently, each P-selective set has small circuits.

Thus the P-selective sets have two faces: They can be so hard as to be undecidable, yet only a hair's breadth of information keeps them from being in P.

Beautiful structures often attract much attention, and that has certainly been the case with the P-selective sets. Much of that attention has been devoted to generalizing and extending concepts and results from Alan's seminal papers. For example, although Alan showed that no NP-hard set (i.e., no set NP-hard with respect to many-one polynomial-time reductions) can be P-selective unless $P = NP$, there followed an intense and productive research line to see whether that claim could be extended beyond many-one reductions to more flexible types of reductions such as bounded-truth-table and truth-table reductions, see [BKS95, Ogi95, AA96], and whether Alan's just-stated result analogously extends to other complexity classes, and what holds for nondeterministic variants of selectivity. See [HT03] for a survey of work along those lines and more generally for a survey of the broad research stream—including important later work by Alan—launched by Alan's beautiful structure known as P-selectivity. As a final comment, in Section 3.3 of this article we'll soon see how selectivity theory surprisingly resolved an important, yet seemingly unrelated, question about removing the ambiguity of nondeterministic functions.

3.2 Much Ado about Functions: Decision Problems Are Not the Only Game in Town

As everyone knows, theoretical computer science is largely built around the complexity of decision problems. After all, NP is (at least if one stays away from certain textbooks!) a class of decision problems. How could there be anything wrong with this approach? After all, satisfiability for example clearly has the property that its search and its decision versions are polynomial-time Turing interreducible.

Alan is one of the people who from the very beginning recognized the tremendous importance of studying functions directly. There are many compelling reasons to study functions directly. Historically, Alan was probably primarily motivated by the extremely interesting role and richness of nondeterministic function classes—and his work there was seminal—and perhaps also by his expertise in one-way functions.

Additionally, experts such as Alan have always known that even for deterministic classes, the justification given above for focusing on decision problems has weaknesses. Satisfiability as used above is not as canonical as it often is. Indeed, it has since been shown that unless $P = NP$ some NP-complete sets are not self-reducible [FO05], which is relevant since

self-reducibility is central in reducing search problems to their natural associated decision problems.¹ But more crucially, Turing reductions are quite powerful, and so may give a blurred view of the actual complexity of the objects involved.

Early on, Alan and his collaborators Book and Long ([BLS84], see also [AM77, BLS85]) introduced a rich range of polynomial-time nondeterministic reductions and studied their properties and relationships. Alan also accessibly spread the word about the importance of taking a function-based view, through his papers on “A Taxonomy of Complexity Classes of Functions” [Sel94] and “Much Ado about Functions” [Sel96] (see also his survey paper on one-way functions [Sel92]).

As he was writing those papers, Alan was also obtaining new insights into search (a type of function version of problems) versus decision, e.g., his paper “P-Selective Sets and Reducing Search to Decision vs. Self-Reducibility” [HNOS96a] (and yes, there is P-selectivity playing a role again!), and he also interestingly studied the role of functions as outputs of oracles [FHOS97].

These days, function versions of problems play such a large role that it is easy to forget that thirty years ago it wasn’t yet clear that that would ever be the case. Alan’s pioneering stress on functions, quirky for its time, was quite prescient.

3.3 Selectivity and Functions Team Up: Is Finding All Solutions Easier than Finding One Solution?

Is finding all solutions to a problem easier than finding one solution? One is tempted to answer: Never! After all, if one can find all solutions, then one can simply take (for example) the lexicographically smallest solution, and one has found one solution (if one exists).

Surprisingly, the reasoning just used falls apart in a nondeterministic context. Let us see this. And then let us see what Alan did about this through bringing together the theory of functions with a generalization of the theory of P-selectivity.

Consider a nondeterministic polynomial-time Turing machine that on each path either rejects or accepts. We will consider the machine to be computing a partial, multivalued function (which in mathematics one would probably call a relation, but in complexity theory the term function is often used even for such multivalued objects). Namely, each rejecting path isn’t considered to contribute anything to the function. But whatever string is on the first work tape of the machine on nondeterministic paths that accept is considered to be an output of the function on the given input. This notion is what is called an NPMV function (a nondeterministic polynomial-time multivalued function). The class NPMV was introduced by Book, Long, and Selman [BLS84] in their seminal work on nondeterministic functions.

Let us make the nature of NPMV clear by giving a very important example. Consider the nondeterministic polynomial-time Turing machine (NPTM) that on input f , interprets f as a boolean formula, nondeterministically guesses an assignment for the variables of f

¹We say “natural” since for any NP-like search problem one can build some decision problem to which it Turing-reduces. But that is not the focus here.

and writes that assignment on its first work tape, and then deterministically uses its other tapes to check whether the assignment satisfies f , and if it does the machine (on that non-deterministic computation path) accepts and if not the machine (on that nondeterministic computation path) rejects. Notice that this machine on input f outputs all satisfying assignments of f . That set could be exponentially large, but that isn't a problem here—the output set is in effect distributed among all the paths, and so it isn't ever stuffed into a single, giant output string. This function is called f_{SAT} .

An NPSV function (a nondeterministic polynomial-time single-valued function) is exactly the same notion as an NPMV function, except NPSV functions must in addition satisfy the property that on each input the cardinality of the output set is at most one. So if no path accepts, that is fine, as the cardinality of the output set is zero. And if one or more paths accept, that is also fine, as long as every one of those accepting paths has precisely the same string on its first work tape, since then that string will be the one and only output.

A central concept in the study of multivalued functions is the notion of a refinement. f_2 refines f_1 exactly if on each input x ,

1. f_2 outputs at least one value if and only if f_1 outputs at least one value, and
2. every output of f_2 is an output of f_1 .

We saw above that f_{SAT} is an NPMV function. But does it have an NPSV refinement? Put another way: As mentioned above, it is easy for NP function-computing machines to find all solutions of an input boolean formula. But can an NP function-computing machine find *one* solution of an input boolean formula (when one exists)? This question is asking whether there is an NPTM that for unsatisfiable formulas has all its paths reject, and for each satisfiable formula has at least one path that computes a satisfying assignment (and accepts) and every accepting path must compute the same satisfying assignment as every other accepting path.

It will now be clear why the trick that works in the deterministic case seems unhelpful here. In the deterministic case we took all the solutions and output the lexicographically smallest. But for an NPMV function to do that, at least in the most obvious way, a path would have to be able to figure out whether the value it would like to output is such that every other path that would like to output a value would in fact like to output a lexicographically equal-or-larger value (and in that case our path would go ahead and output its value, and otherwise would kill itself off by rejecting). But figuring *that* out seems to take an extra quantification that our machine doesn't have in its arsenal. Big picture, NPSV is in some sense asking for a strong degree of coordination among paths that have no way of communicating with each other.

Of course, the previous paragraph is just an intuitive handwave, not a proof. Proving that NPMV functions don't all have NPSV refinements (unless the polynomial hierarchy collapses) required a surprising twist. Alan and his collaborators had developed nondeterministic analogues of P-selectivity [HHN⁺95, HNOS96b]. Although at first one might think that selectivity has little to do with NPMV and NPSV functions, nondeterministic

selectivity and its connection with nonuniform classes such as $(\text{NP} \cap \text{coNP})/\text{poly}$ turned out to be exactly the tool Alan and his collaborators needed to prove that if f_{SAT} has an NPSV refinement (equivalently, if every NPMV function has an NPSV refinement), then the polynomial hierarchy collapses to NP^{NP} [HNOS96b].

A collapse of the polynomial hierarchy to NP^{NP} still can be obtained even if one merely assumes that one can refine at-most-2-valued NPMV functions to NPSV functions [HNOS96b]. Later work about nonuniform classes let one conclude from Alan’s work a slightly more extensive collapse in both these cases [CCHO05], in particular collapsing the polynomial hierarchy to $\text{S}_2^{\text{NP} \cap \text{coNP}}$. Alan and his collaborators also explored the question of, for $k \geq 2$, what collapses would occur if one could reduce at-most- $(k + 1)$ -valued NPMV functions to at-most- k -valued NPMV functions, and obtained polynomial-hierarchy collapses to NP^{NP} for each such case [NRRS98]. Strengthening those collapses to a collapse to $\text{S}_2^{\text{NP} \cap \text{coNP}}$ remains open to this day and is a quite interesting challenge that has stymied many a graduate student. See also [HOW02] for some cases where refinements in fact are possible.

In summary, Alan didn’t define nondeterministic functions and selectivity theory for the purpose of shedding light on whether one could reduce many solutions to one solution. However, his notions were so beautiful and flexible that they were central in the resolution of that issue.

3.4 Positive Relativization: Getting Real-World Consequences from Oracles

Informally put, relativizing by an oracle A means giving all machines unit-cost access to A , i.e., machines can as often as they like write a string onto a query tape and immediately the machine will be told whether that string is a member of A . In some sense, relativization changes the universe’s “ground rules” about what information is available to computing machines, but does so fairly—all machines now have access to A .

Experienced complexity theorists may be a bit surprised by what we’ve chosen as our example of a beautiful contribution by Alan to the theory of relativization.

After all, one of Alan’s earliest works is the famous 1979 paper “A Second Step Toward the Polynomial Hierarchy” with Baker [BS79], which showed that there is an oracle relative to which the second and third levels of the polynomial hierarchy separate. The proof is extremely clever and powerful—employing what one might dub a nested double contradiction architecture. How amazing that proof was is made clear by how long it took to take the “third step”; that didn’t occur until the work many years later of Yao [Yao85] and Håstad [Hås87], which drew on different techniques and separated the entire polynomial hierarchy.

Structural complexity theorists will remember well the important paper by Homer and Selman [HS92] that built a relativized world in which all NP^{NP} -complete sets were polynomial-time isomorphic. Not too many years later, Fenner, Fortnow, and Kurtz [FFK96], surely encouraged by that paper, obtained the same result for NP itself, thus directly speaking to the relativized version of Berman and Hartmanis’s Isomorphism

Conjecture [BH77]. Rogers [Rog97] even put a great cherry on top of that, by obtaining isomorphism while not killing off one-way functions—and we’ll speak more about those in Section 3.5, since Alan’s work on them is seminal.

But as beautiful and important as those results of Alan’s are, at least as beautiful, if perhaps less well known these days, is the pioneering work by Alan and his collaborators on positive relativization.

What is the biggest perceived weakness of relativization theory? It probably is that it often seems that one can do almost anything in relativized worlds, and that doing so has little connection with the real world. For example, Baker, Gill, and Solovay [BGS75] famously showed that there are oracles making P equal to NP , and that there are oracles making P not equal to NP . Does this resolve the P versus NP problem in the real world? Fat chance.

Those oracles actually do tell us something quite important, namely, they tell us that no proof technique that relativizes can prove either $P = NP$ or $P \neq NP$. And so there has been quite a bit of research regarding proof techniques such as arithmetization (see [BF91]) and results such as $IP = PSPACE$ [Sha92] that seem not to relativize [FS88]. An interesting perspective on this is given by Hartmanis et al. [HCC⁺92], who argue that there have been nonrelativizing proof techniques for a long time; see also [CCG⁺94].

In any case, the fact that relativization results such as those of Baker, Gill, and Solovay [BGS75] fail to resolve the analogous real-world problems is often presented as a clear weakness of studying oracles.

Positive relativization provides an utterly lovely response to this weakness: *We should find cases where obtaining an oracle result would imply real-world results.* Positive relativization as such was introduced and explored by Alan, Ron Book, and their collaborators ([SXB83], see also [Boo81, BLS84, BBL⁺84, BLS85, LS86, Boo87, Boo89]).

Let us illustrate positive relativization by one of its most striking examples. (Note: What we here speak of as “positive relativization” also sweeps in what Book [Boo89] distinguishes as “negative relativization.”) Suppose we claim that there is sparse oracle relative to which the polynomial hierarchy collapses. Are you thrilled? Perhaps not, since that might not say anything about the real world. Suppose we claim that there is sparse oracle relative to which the polynomial hierarchy is infinite. Now are you thrilled? Perhaps again not, since that might not say anything about the real world. You’re seeming pretty hard to thrill, my friend.

But wait. These things *would* say something about the real world. That is because Balcázar, Book, Long, Schöning, and Selman [BBL⁺84, BBS86, LS86] proved the following result.

Theorem 3.1 1. *The polynomial hierarchy collapses if and only if there is a sparse oracle relative to which the polynomial hierarchy collapses.*
 2. *The polynomial hierarchy is infinite if and only if there is a sparse oracle relative to which the polynomial hierarchy is infinite.*

This theorem tells us we should care, and indeed be thrilled, if someone has a sparse oracle that makes the polynomial hierarchy infinite, or has a sparse oracle that makes the polynomial hierarchy collapse. Indeed, that person—thanks to the groundwork of Alan just presented—will have so changed the real-world landscape of complexity that he or she is sure to win a Turing Award.

So positive relativization links oracle results to real-world collapses and separations. And that is a lovely thing to do. By now, a broad range of positive relativizations are known, involving issues ranging from sparse sets (as above) to tally sets (historically the earliest form of what now is called positive relativization), to number of queries, to aspects of the form and structure of the querying. A cynic might say that such results just tell us which oracle results are too hard to hope to ever get, since they would resolve major real-world issues. But an optimist might say that these offer an extra potential path to resolve those major real-world issues.

The path isn't a universal one. For example, Alan's student Roy Rubinfeld writing "On the Limitations of Positive Relativization" [HR92] showed that for "semantic" classes such as UP, R, and $\text{NP} \cap \text{coNP}$, positive-relativization attempts with tally oracles fail even though the analogous positive-relativization results are well known to succeed for NP and the polynomial hierarchy's levels.

But even if not universal, the path of positive relativization is certainly a beautiful insight, especially since it has been broadly applied to the centrally important class NP.

3.5 One-Way Functions: A Complexity-Theoretic Characterization of Their Existence

For the final of our five beautiful structures sculpted by Alan, we take the rigorous definition of the notion of a (complexity-theoretic) one-way function, and the complete characterization (in terms of the collapse and noncollapse of important complexity classes) of whether one-way functions exist. Alan did this work with his first Ph.D. student, Joachim Grollmann [GS88], and this was also achieved independently by Ko [Ko85] (see also [Ber77]).

Many readers have probably already seen the definition and theorem alluded to above, since they can be found in such excellent complexity texts as for example Papadimitriou [Pap94] and Du and Ko [DK00]. But since it is well worth everyone knowing, let us quickly give the definition and the theorem.²

A (total) function f is a (complexity-theoretic) one-way function if and only if

1. f is polynomial-time computable,

²These frame and explore the notion of complexity-theoretic one-way functions. More demanding definitions of one-way functions—focusing on the success probability of probabilistic inverters—are important in cryptography. It would be wonderful to prove that such extremely strong one-way functions exist. However, even the easier step of proving that complexity-theoretic one-way functions exist would—by the characterization result soon to be stated and the fact that $\text{UP} \subseteq \text{NP}$ —separate P from NP and win the prover a million-dollar Clay Mathematics Institute prize [Cla]. So achieving the "easier" step is quite important although surely not easy.

2. f is polynomially honest (i.e., there is a polynomial q such that for each x , $q(|f(x)|) \geq |x|$; informally put, f doesn't shrink its inputs more than polynomially much),
3. f is an injective (i.e., one-to-one) function, and
4. f is not polynomial-time invertible (since we're concerned here only with total, injective functions, we may use as our definition of this that for any polynomial-time function g , there exists an x such that $g(f(x)) \neq x$).

Valiant's [Val76] class UP is the class of all NP sets that are accepted by some NPTM that on no input has more than one accepting path.

The characterization theorem that brings together complexity classes and the existence of one-way functions is the following: One-way functions exist if and only if $P \neq UP$ [Ko85, GS88]. The proof of this theorem elegantly goes back and forth between the world of one-way functions and the world of nondeterministic Turing machines.

This concept and characterization naturally inspired much related work. (See [HO02, Chapter 2: "The One-Way Function Technique"] for a survey-like treatment, including proofs, of the characterization mentioned above and the related results mentioned in this paragraph.) For example, researchers have looked at (what in effect is the study of) k -bounded-ambiguity one-way functions (which interestingly enough stand or fall together with Alan's notion of one-way functions, thanks to a nice induction proof of Watanabe [Wat88]), and for multi-argument one-way functions one can study algebraic properties such as associativity and commutativity (but again such functions turn out to stand or fall together with Alan's notion of one-way functions [RS97, HR99]). All such work is clearly indebted to the beautiful, seminal work of Alan, his student Grollmann, and Ko.

4 Conclusion

Anyone who has read Alan's articles knows that Alan always knows the exact right number of words to use to motivate, explain, and develop a concept. So to conclude, let us try to take a page (or a quarter of a page) from Alan and keep things as focused as possible, so that the points this article has been trying to make speak clearly: Alan's work stands out as extraordinary in that it has introduced and powerfully explored a tremendous number of utterly beautiful structures. We spoke at the start of the article about the elusive tapestry of coherent beauty that binds together the structure of computation. Alan's career has been very successfully devoted to revealing parts of that tapestry. Beyond that, Alan has been boundlessly generous and inspirational to his many younger collaborators and has been a leader in service to the field. So with warmest thanks for so very much, and comforted by the knowledge that Alan intends to keep his hand in the field, let us wish Alan the most wonderful of retirements.

References

- [AA96] M. Agrawal and V. Arvind. Quasi-linear truth-table reductions to P-selective sets. *Theoretical Computer Science*, 158(1–2):361–370, 1996.
- [AM77] L. Adleman and K. Manders. Reducibility, randomness, and intractibility. In *Proceedings of the 9th ACM Symposium on Theory of Computing*, pages 151–153. ACM Press, May 1977.
- [BBL⁺84] J. Balcázar, R. Book, T. Long, U. Schöning, and A. Selman. Sparse oracles and uniform complexity classes. In *Proceedings of the 25th IEEE Symposium on Foundations of Computer Science*, pages 308–313. IEEE Computer Society Press, October 1984.
- [BBS86] J. Balcázar, R. Book, and U. Schöning. The polynomial-time hierarchy and sparse oracles. *Journal of the ACM*, 33(3):603–617, 1986.
- [Ber77] L. Berman. *Polynomial Reducibilities and Complete Sets*. PhD thesis, Cornell University, Ithaca, NY, 1977.
- [BF91] L. Babai and L. Fortnow. Arithmetization: a new method in structural complexity theory. *Computational Complexity*, 1(1):41–66, 1991.
- [BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the P=?NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.
- [BH77] L. Berman and J. Hartmanis. On isomorphisms and density of NP and other complete sets. *SIAM Journal on Computing*, 6(2):305–322, 1977.
- [BKS95] R. Beigel, M. Kummer, and F. Stephan. Approximable sets. *Information and Computation*, 120(2):304–314, 1995.
- [BLS84] R. Book, T. Long, and A. Selman. Quantitative relativizations of complexity classes. *SIAM Journal on Computing*, 13(3):461–487, 1984.
- [BLS85] R. Book, T. Long, and A. Selman. Qualitative relativizations of complexity classes. *Journal of Computer and System Sciences*, 30(3):395–413, 1985.
- [Boo81] R. Book. Bounded query machines: On NP and PSPACE. *Theoretical Computer Science*, 15:27–39, 1981.
- [Boo87] R. Book. Towards a theory of relativizations: Positive relativizations. In *Proceedings of the 4th Annual Symposium on Theoretical Aspects of Computer Science*, pages 1–21. Springer-Verlag *Lecture Notes in Computer Science* #247, 1987.

- [Boo89] R. Book. Restricted relativizations of complexity classes. In J. Hartmanis, editor, *Computational Complexity Theory*, pages 47–74. American Mathematical Society, 1989. Proceedings of Symposia in Applied Mathematics #38.
- [Bra79] G. Brassard. A note on the complexity of cryptography. *IEEE Transactions on Information Theory*, 25(2):232–233, 1979.
- [BS79] T. Baker and A. Selman. A second step toward the polynomial hierarchy. *Theoretical Computer Science*, 8:177–187, 1979.
- [CCG⁺94] R. Chang, B. Chor, O. Goldreich, J. Hartmanis, J. Håstad, and D. Ranjan. The Random Oracle Hypothesis is false. *Journal of Computer and System Sciences*, 49(1):24–39, 1994.
- [CCHO05] J. Cai, V. Chakaravarthy, L. Hemaspaandra, and M. Ogihara. Competing provers yield improved Karp–Lipton collapse results. *Information and Computation*, 198(1):1–23, 2005.
- [Cla] Clay Mathematics Institute. www.claymath.org/millennium-problems. URL verified June 2014.
- [DK00] D. Du and K. Ko. *Theory of Computational Complexity*. John Wiley and Sons, 2000.
- [ESY84] S. Even, A. Selman, and Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 61(2):159–173, 1984.
- [FFK96] S. Fenner, L. Fortnow, and S. Kurtz. The Isomorphism Conjecture holds relative to an oracle. *SIAM Journal on Computing*, 25(1):193–206, 1996.
- [FHOS97] S. Fenner, S. Homer, M. Ogiwara, and A. Selman. Oracles that compute values. *SIAM Journal on Computing*, 26(4):1043–1065, 1997.
- [FO05] P. Faliszewski and M. Ogihara. Separating the notions of self- and autoreducibility. In *Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science*, pages 308–315. Springer-Verlag *Lecture Notes in Computer Science* #3618, August/September 2005.
- [FS88] L. Fortnow and M. Sipser. Are there interactive protocols for co-NP? *Information Processing Letters*, 28:249–251, 1988.
- [GHJY91] J. Goldsmith, L. Hemachandra, D. Joseph, and P. Young. Near-testable sets. *SIAM Journal on Computing*, 20(3):506–523, 1991.
- [GHSW14] C. Glaßer, A. Hughes, A. Selman, and N. Wisiol. Disjoint NP-pairs and propositional proof systems. *SIGACT News*, 45(4), 2014. In Preparation.

- [GNR⁺13] C. Glaßer, D. Nguyen, C. Reitwießner, A. Selman, and M. Witek. Autoreducibility of complete sets for log-space and polynomial-time reductions. In *Proceedings of the 40th International Colloquium on Automata, Languages, and Programming, Part I*, pages 473–484, July 2013.
- [GOP⁺09] C. Glaßer, M. Ogihara, A. Pavan, A. Selman, and L. Zhang. Autoreducibility and mitocity. *SIGACT News*, 40(3):61–76, 2009.
- [GS88] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17(2):309–335, 1988.
- [Hås87] J. Håstad. *Computational Limitations of Small-Depth Circuits*. MIT Press, 1987.
- [HCC⁺92] J. Hartmanis, R. Chang, S. Chari, D. Ranjan, and P. Rohatgi. Relativization: A revisionistic retrospective. *Bulletin of the EATCS*, 47:144–153, 1992.
- [HH91] L. Hemachandra and A. Hoene. On sets with efficient implicit membership tests. *SIAM Journal on Computing*, 20(6):1148–1156, 1991.
- [HHN⁺95] L. Hemaspaandra, A. Hoene, A. Naik, M. Ogiwara, A. Selman, T. Thierauf, and J. Wang. Nondeterministically selective sets. *International Journal of Foundations of Computer Science*, 6(4):403–416, 1995.
- [HNOS96a] E. Hemaspaandra, A. Naik, M. Ogihara, and A. Selman. P-selective sets and reducing search to decision vs. self-reducibility. *Journal of Computer and System Sciences*, 53(2):194–209, 1996.
- [HNOS96b] L. Hemaspaandra, A. Naik, M. Ogihara, and A. Selman. Computing solutions uniquely collapses the polynomial hierarchy. *SIAM Journal on Computing*, 25(4):697–708, 1996.
- [HO02] L. Hemaspaandra and M. Ogihara. *The Complexity Theory Companion*. Springer-Verlag, 2002.
- [HOW02] L. Hemaspaandra, M. Ogihara, and G. Wechsung. Reducing the number of solutions of NP functions. *Journal of Computer and System Sciences*, 64(2):311–328, 2002.
- [HPRS12] A. Hughes, A. Pavan, N. Russell, and A. Selman. A thirty year old conjecture about promise problems. In *Proceedings of the 39th International Colloquium on Automata, Languages, and Programming, Part I*, pages 473–484. Springer-Verlag *Lecture Notes in Computer Science* #7391, July 2012.
- [HR92] L. Hemachandra and R. Rubinfeld. Separating complexity classes with tally oracles. *Theoretical Computer Science*, 92(2):309–318, 1992.

- [HR99] L. Hemaspaandra and J. Rothe. Creating strong, total, commutative, associative one-way functions from any one-way function in complexity theory. *Journal of Computer and System Sciences*, 58(3):648–659, 1999.
- [HS92] S. Homer and A. Selman. Oracles for structural properties: The isomorphism problem and public-key cryptography. *Journal of Computer and System Sciences*, 44(2):287–301, 1992.
- [HS97] L. Hemaspaandra and A. Selman, editors. *Complexity Theory Retrospective II*. Springer-Verlag, 1997.
- [HS11] S. Homer and A. Selman. *Computability and Complexity Theory*. Springer-Verlag, 2nd edition, 2011.
- [HT03] L. Hemaspaandra and L. Torenvliet. *Theory of Semi-Feasible Algorithms*. Springer-Verlag, 2003.
- [Joc68] C. Jockusch. Semirecursive sets and positive reducibility. *Transactions of the AMS*, 131(2):420–436, 1968.
- [KL80] R. Karp and R. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proceedings of the 12th ACM Symposium on Theory of Computing*, pages 302–309. ACM Press, April 1980. An extended version has also appeared as: Turing machines that take advice, *L’Enseignement Mathématique*, 2nd series, 28:191–209, 1982.
- [Ko83] K. Ko. On self-reducibility and weak P-selectivity. *Journal of Computer and System Sciences*, 26(2):209–221, 1983.
- [Ko85] K. Ko. On some natural complete operators. *Theoretical Computer Science*, 37(1):1–30, 1985.
- [LLS75] R. Ladner, N. Lynch, and A. Selman. A comparison of polynomial time reducibilities. *Theoretical Computer Science*, 1(2):103–124, 1975.
- [Lon82] T. Long. Strong nondeterministic polynomial-time reducibilities. *Theoretical Computer Science*, 21(1):1–25, 1982.
- [LS86] T. Long and A. Selman. Relativizing complexity classes with sparse oracles. *Journal of the ACM*, 33(3):618–627, 1986.
- [MP79] A. Meyer and M. Paterson. With what frequency are apparently intractable problems difficult? Technical Report MIT/LCS/TM-126, Laboratory for Computer Science, MIT, Cambridge, MA, 1979.

- [MS72] A. Meyer and L. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *Proceedings of the 13th IEEE Symposium on Switching and Automata Theory*, pages 125–129. IEEE Press, October 1972.
- [NRRS98] A. Naik, J. Rogers, J. Royer, and A. Selman. A hierarchy based on output multiplicity. *Theoretical Computer Science*, 207(1):131–157, 1998.
- [NS14] D. Nguyen and A. Selman. Non-autoreducible sets for NEXP. In *Proceedings of the 31st Annual Symposium on Theoretical Aspects of Computer Science*, pages 590–601, March 2014.
- [Ogi95] M. Ogiwara. Polynomial-time membership comparable sets. *SIAM Journal on Computing*, 24(5):1068–1081, 1995.
- [OW91] M. Ogiwara and O. Watanabe. On polynomial-time bounded truth-table reducibility of NP sets to sparse sets. *SIAM Journal on Computing*, 20(3):471–483, June 1991.
- [Pap94] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- [Rog97] J. Rogers. The Isomorphism Conjecture holds and one-way functions exist relative to an oracle. *Journal of Computer and System Sciences*, 54(3):412–423, 1997.
- [RS97] M. Rabi and A. Sherman. An observation on associative one-way functions in complexity theory. *Information Processing Letters*, 64(5):239–244, 1997.
- [Sch83] U. Schöning. A low and a high hierarchy within NP. *Journal of Computer and System Sciences*, 27(1):14–28, 1983.
- [Sch86] U. Schöning. Complete sets and closeness to complexity classes. *Mathematical Systems Theory*, 19(1):29–42, 1986.
- [Sel74] A. Selman. On the structure of NP. *Notices of the AMS*, 21(5):A–498, 1974. Erratum in the same journal, 21(6):310.
- [Sel78] A. Selman. Polynomial time enumeration reducibility. *SIAM Journal on Computing*, 7(4):440–457, 1978.
- [Sel79] A. Selman. P-selective sets, tally languages, and the behavior of polynomial time reducibilities on NP. *Mathematical Systems Theory*, 13(1):55–65, 1979.
- [Sel81] A. Selman. Some observations on NP real numbers and P-selective sets. *Journal of Computer and System Sciences*, 23(3):326–332, 1981.
- [Sel82a] A. Selman. Analogues of semirecursive sets and effective reducibilities to the study of NP complexity. *Information and Control*, 52(1):36–51, 1982.

- [Sel82b] A. Selman. Reductions on NP and P-selective sets. *Theoretical Computer Science*, 19(3):287–304, 1982.
- [Sel88] A. Selman. Natural self-reducible sets. *SIAM Journal on Computing*, 17(5):989–996, 1988.
- [Sel90] A. Selman, editor. *Complexity Theory Retrospective*. Springer-Verlag, 1990.
- [Sel92] A. Selman. A survey of one-way functions in complexity theory. *Mathematical Systems Theory*, 25(3):203–221, 1992.
- [Sel94] A. Selman. A taxonomy of complexity classes of functions. *Journal of Computer and System Sciences*, 48(2):357–381, 1994.
- [Sel96] A. Selman. Much ado about functions. In *Proceedings of the 11th Annual IEEE Conference on Computational Complexity*, pages 198–212. IEEE Computer Society Press, May 1996.
- [Sha92] A. Shamir. $IP = PSPACE$. *Journal of the ACM*, 39(4):869–877, 1992.
- [Sto76] L. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.
- [SXB83] A. Selman, Xu M.-R., and R. Book. Positive relativizations of complexity classes. *SIAM Journal on Computing*, 12(3):565–579, 1983.
- [Val76] L. Valiant. The relative complexity of checking and evaluating. *Information Processing Letters*, 5(1):20–23, 1976.
- [Wat88] O. Watanabe. On hardness of one-way functions. *Information Processing Letters*, 27(3):151–157, 1988.
- [Yao85] A. Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science*, pages 1–10. IEEE Computer Society Press, October 1985.